REMARKS

Claims 7, 9-11, 13, 15, 18-20, and 22-25 are pending. Claim 7 stands rejected under 35 U.S.C. § 112, ¶ 2 as being indefinite for failing to particularly point out and distinctly claim the subject matter which the applicant regards as the invention. Claims 15, 18-20, and 22 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,125,391 to Meltzer et al. in view of U.S. Patent No. 6,622,170 to Harrison et al. Claims 7, 9-11, and 13 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,125,391 to Meltzer et al. in view of U.S. Patent No. 6,622,170 to Harrison et al. and Official Notice. Claims 23-25 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,125,391 to Meltzer et al. in view of U.S. Patent No. 6,622,170 to Harrison et al. and NetWare Directory Services by Frank.

Reconsideration is requested. No new matter is added. The specification is amended. Claims 7, 11, 15, 18, 22, and 22 are amended. Claims 10, 13, and 23-25 are canceled. Claims 26-41 are added. The rejections are traversed. Claims 7, 9, 11, 15, 18-20, 22, and 26-41 remain in the case for consideration. The Examiner is requested to treat any changes to the claims not reflected as strikethrough or underline text as inadvertent typographical errors.

Because of the complexity of the amendments presented, a clean version of the claims is attached to this Response, for the Examiner's convenience.

REJECTION OF CLAIMS UNDER 35 U.S.C. § 103(a)

Referring to claim 7, the invention is directed toward a software program facilitating the use of a distributed directory running in a computer network, the program comprising being stored on a recordable medium and including instructions for: receiving an event from the distributed directory into an XML generator; converting the event into XML data representing the event; transforming the XML data representing the event to a first predetermined formed by a transformation processor using a first stylesheet, the first predetermined format being responsive to a first application running in the computer network; transmitting the transformed XML data representing the first event to the first application; transforming the XML data representing the event to a second predetermined formed by the transformation processor using a second stylesheet, the second predetermined format being responsive to a second application running in the computer network.

Referring to claim 15, the invention is directed toward a software program for facilitating the use of a distributed directory running in a computer network, the program comprising instructions for: receiving a first event from a first application in a first native

application format; converting the first event into markup language data; transforming the first event to a predetermined format by a transformation processor using a transformation profile, the predetermined format being responsive to the distributed directory, the transformation profile including formatting instructions for transforming the markup language data to the predetermined format; transmitting the transformed first event to the distributed directory; receiving a second event from a second application in a second native application format; converting the second event into markup language data; transforming the second event to the predetermined format by the transformation processor using the transformation profile; transmitting the transformed second event to the distributed directory.

Referring to claim 18, the invention is directed toward a distributed computer system comprising: a first processor connected to a network for executing computer code; a second processor connected to the network for executing computer code; a first memory connected to the first processor; a second memory connected to the second processor; a distributed directory, wherein first and second portions of the distributed directory are stored in the first memory and the second memory, respectively; a first application, a portion of which being stored in one of the first memory and the second memory; a second application, a portion of which being stored in one of the first memory and the second memory; a first transformation profile defining a first predetermined format for use by the first application; a second transformation profile defining a second predetermined format for use by the second application; software for detecting a directory event in the distributed directory; software for transforming the directory event to the first predetermined format by using a generic transformation tool and the first transformation profile; software for transforming the directory event to the second predetermined format by using the generic transformation tool and the second transformation profile; software for providing to the first application the directory event transformed to the first predetermined format; and software for providing to the second application the directory event transformed to the second predetermined format.

Referring to claim 26, the invention is directed toward a driver infrastructure for interfacing a directory and applications comprising: a generator to receive a directory event from the directory and to generate a generic data for the directory event; a first transformation profile defining a first predetermined format for use by a first application; a second transformation profile defining a second predetermined format for use by a second application; a transformation processor to transform the generic data for the directory event into a first application data for the first application using the first transformation profile and to transform the generic data for the directory event into a second application data for the

second application using the second transformation profile; and a transmitter to transmit the first application data to the first application and to transmit the second application data to the second application.

Referring to claim 33, the invention is directed toward a method for interfacing with a directory in a computing system, comprising: providing a first transformation profile defining a first predetermined format for use by a first application; providing a second transformation profile defining a second predetermined format for use by a second application; detecting an event in the directory; transforming the event to the first predetermined format by using a transformation tool and the first transformation profile; transforming the event to the second predetermined format by using the transformation tool and the second transformation profile; providing to the first application the event transformed to the first predetermined format; and providing to the second application the event transformed to the second predetermined format.

In contrast, Meltzer teaches market makers using documents in trading partner network. As shown in FIG. 15, Meltzer receives documents via a communications agent. The documents can be in any syntax. Meltzer converts the documents to XML, then uses an XML parser to verify that the documents are properly formatted in XML. Using the business interface definition compiler (BIDC), the documents are compiled into Java documents. The Java documents are then delivered to the enterprise solutions using the document service.

In rejecting claims 15 and 18, the Examiner has argued that both the distributed directory *and* the applications can be analogized to the source of the documents in FIG. 15 of Meltzer. This concept is not realistic. If the analogy is to hold, then *exactly* one end of FIG. 15 of Meltzer must be analogous to the distributed directory, and exactly one end is analogous to the applications. And regardless of which end the Examiner were to designate as analogous to the distributed directory and which were analogous to the applications, the analogy is flawed.

Independent claims 7, 18, 32, and 40 (and the claims dependent from these independent claims) describe transformations from the distributed directory to multiple applications. If the distributed directory is analogous to the left end of FIG. 15 (that is, the source of the documents), then the analogy fails because Meltzer teaches transforming the data into only a single other format. In FIG. 15, the *only* thing being transmitted between the XML processor and the document router are the Java Beans. In other words, all of the data is being transformed into the *same* format. But the invention is not limited to all applications receiving data in the same format. Indeed, if such an assumption could be made, then there

would be no need for the transformation to the generic data description (e.g., XML): the data could be transformed *directly* to the format the applications supported. A premise of the invention is that different applications can use different data formats, and so require the data be provided in the different formats. Thus, interpreting the source of the documents in FIG. 15 to be the distributed directory fails for not transforming the data into the different formats needed by the different applications, and claims 7, 9, 11, 18-20, 22, 26-41 would all be allowable.

The Examiner has pointed to column 81, lines 24-43 as teaching the concept of different transformation profiles. The Applicant respectfully disagrees. Column 81, lines 24-43 mention that the compiler can take the data and generate several different "target forms." But Meltzer goes on to recite that this is accomplished using "'tree-to-tree' transformations": that is, one form is then transformed into another. This is quite different from taking the original data and transforming it directly to multiple different forms.

The Examiner also argued that because Meltzer mentioned an XML style sheet that Meltzer taught transformation profiles. But a careful reading of Meltzer makes clear that Meltzer teaches *creating* an XSL style sheet from the data; this is quite different from using a stylesheet to manage the transformation. In Meltzer, the XSL style sheet is an end-product of the transformation; in the invention, the stylesheet is a catalyst of the transformation, and necessarily exists *before* the transformation begins.

The alternative position, that the applications would be analogous to the source of the documents in FIG. 15 of Meltzer, works only slightly better. With the alternative interpretation of Meltzer, the transformations are all going to XML and then to a single format (to that used by the distributed directory). This is similar to the transformation described in independent claim 15. But Meltzer still fails to teach the feature of the transformation profiles. Thus, claim 15 is also allowable.

The Examiner might believe that the Java format of Meltzer is an intermediate format of its own, and that the data is further transformed before being delivered to the ultimate user (be it application or directory). But the advantage of having an intermediate format is that it provides a common point from which all translations can occur. Having two such intermediate formats means that either there is a totally wasted step in Meltzer (that is, the translation from XML to Java, with the possibility of data lost in the additional transformation), or else it is necessary to use two intermediate formats, which means that the claims (which use only one generic format) is not disclosed in Meltzer.

In addition, regardless of which way the Examiner chooses to interpret Meltzer, Meltzer fails to teach the concept of reversing the flow of data. FIG. 15 of Meltzer is explicit in showing data flowing in only one direction: from left to right. There is no data shown flowing in the reverse direction (from right to left), and the accompanying description also fails to mention data flowing in the reverse direction. Independent claims 7, 18, 32, and 40 are directed toward data flowing from the distributed directory to the applications; independent claim 15 is directed toward data flowing from the applications to the distributed directory. Because independent claims 7, 15, 18, 32, and 40 in combination teach data flowing in both directions, the Examiner cannot reject all of these claims based on a reference that teaches data flow in only a single direction. And claims 29 and 41 (dependent from independent claims 18 and 40, respectively) provide for data flowing in *both* directions. Thus, since the direction of data flow in FIG. 15 of Meltzer is unidirectional but the data can flow in both directions in the invention, at least claims 29 and 41 are allowable.
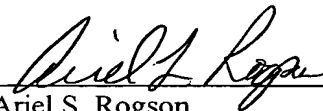
The Examiner's argument that the market maker server node functions as a distributed directory is also incorrect. Meltzer describes the market maker as a server binding services, and translating data formats required by legacy systems. In other words, Meltzer teaches a specialized router and translator. This is a long way from saying that Meltzer manages document storage, as performed by a distributed directory.

In addition, Harrison as a reference teaches *away* from the invention. In the second sentence of the abstract, Harrison says that "[a] current tree of directory information maintained at an LDAP server *is used by LDAP clients*." The application has been clear in that the applications *cannot* interface with the distributed directory. Were the applications able to interface in the manner suggested by Harrison, the invention would not be necessary. Thus, Harrison contradicts a basic premise of the invention, and Meltzer and Harrison cannot be combined as the Examiner suggests.

For the foregoing reasons, reconsideration and allowance of claims 7, 9, 11, 15, 18-20, 22, and 26-41 of the application as amended is solicited. **Because this application is undergoing a second Request for Continued Examiner, the Examiner is requested to telephone the undersigned at (503) 222-3613 to schedule an interview.**

Respectfully submitted,

MARGER JOHNSON & McCOLLOM, P.C.

Ariel S. Rogson

Ariel S. Rogson
Reg. No. 43,054

MARGER JOHNSON & McCOLLOM, P.C.
1030 SW Morrison Street
Portland, OR 97205
503-222-3613
Customer No. 20575

I HEREBY CERTIFY THAT
THIS CORRESPONDENCE IS
BEING DEPOSITED WITH THE
UNITED STATES POSTAL
SERVICE AS FIRST CLASS
MAIL IN AN ENVELOPE
ADDRESSED TO:

☐ COMMISSIONER OF PATENTS
AND TRADEMARKS WASHINGTON
D.C. 20231

☒ MAIL STOP RCE
COMMISSIONER FOR PATENTS
BOX 1450
ALEXANDRIA, VA 22313-1450

☐ BOX
COMMISSIONER
FOR TRADEMARKS 2900 CRYSTAL
DRIVE ARLINGTON, VA 22202-3513

ON: